Сумський державний педагогічний університет імені А.С.Макаренка

Фізико-математичний факультет

Кафедра інформатики

Нищета Микола

МЕТОДИЧНІ ОСОБЛИВОСТІ НАВЧАННЯ ПРОГРАМУВАННЮ УЧНІВ 8 КЛАСУ

Спеціальність: 014 Середня освіта (Інформатика)
Галузь знань: 01 Освіта/Педагогіка
Кваліфікаційна робота на здобуття освітнього ступеня бакалавра

Науковий керівник:
_____Н.В. Дегтярьова,
кандидат педагогічних наук, доцент,
доцент кафедри інформатики
«____» _____ 2021 року
Виконавець:
_____ М. Нищета
«____» _____ 2021 року

Суми 2021

ЗМІСТ

ВСТУП

Яку сферу життєдіяльності людини ми б не взяли: маркетинг, політику, медицину, освіту, — без залучення комп’ютерних технологій в сучасному світі нічого не обходиться. Для кожної з цих сфер розробляються відповідні програми. Отже, сьогодні спеціалісти ІТ – галузі є затребуваними. А саме, дедалі популярнішими стають спеціалісти в сфері розробки та просування сайтів, фахівці з напрямку кібербезпеки, штучного інтелекту, віртуальної реальності та інші. Завдяки таким спеціалістам створюються веб-сторінки, забезпечується цілісність даних та їх безпека, розробляється різноманітне програмне забезпечення. Кожен фахівець повинен володіти такими якостями: здатність швидко навчатись, аналітичний склад розуму. Щодо професійних навичок, актуальними є: володіння технічною англійською мовою не нижче Upper Intermediate, знання мов програмування (найчастіше це: Java, C, C++, JavaScript, PHP, Python тощо). Щоб оволодіти даними якостями, вміннями та навичками необхідно починати як можна раніше вивчати основи програмування.

Нагальною проблемою стало вивчення шкільного курсу інформатики, зокрема питань навчання програмування у школі. Дослідники, які розглядають програмування як спосіб спілкування з комп'ютером на зрозумілій йому мові, завжди підтримували ідею навчання учнів програмування з раннього віку [1]. В Україні учні 8 класу мають засвоювати деякі елементарні знання з цієї дисципліни, згідно з навчальною програмою Міністерства освіти і науки України. А саме тему, «Алгоритми та програми». Учні засвоюють базові поняття та виконують елементарні задачі з програмування[2]. За допомогою такого середовища, як "Scratch", вчителі інформатики готують учнів до навчання алгоритмізації і програмування якісно, доступно і легко на практичних прикладах, підвищивши при цьому рівень мотивації учнів до навчання за рахунок використання різних мультимедійних можливостей середовища та ігрових компонентів. А завдяки ознайомленню учня з базовими

поняттями, принципами алгоритмізації, полегшується подальше знайомство з мовами програмування, які без попередньої підготовки освоїти складно, зокрема і Python.

З впевненістю можна сказати, що кожне наступне покоління більш прогресивне, ніж попереднє. Одним з факторів, що сприяють такому стану речей, є інформатизація суспільства.

Адаптація до змін в сучасному суспільстві супроводжується змінами в системі освіти. Одним з позитивних змін для школярів є включення мови Python, з 2019 року в шкільні олімпіади.

Актуальність теми дослідження обумовлена безліччю факторів, найважливішими з яких є:

- вимоги МОН до знаннєвої складової, де розуміння призначення мови програмування є одною з складових;
- формування готовності учня до усвідомленого вибору професії в сфері ІТ-технологій;
- формування готовності учнів до саморозвитку та здатності до самоосвіти, яке згідно МОН є однією з найважливіших цілей сучасної освіти;
- шкільний курс програмування є складним для учнів, тому існує необхідність подавати його доступніше.

Проблема дослідження полягає в необхідності вирішення протиріччя: з одного боку пов'язаної з надзвичайною актуальністю використання мови Python, а з іншого боку, з недоліком методичних розробок до вивчення програмування на мові Python в курсі шкільної інформатики.

Тому **метою дослідження** є вивчення методичних особливостей навчання мови програмування Python в курсі шкільної інформатики 8 класу.

Об'єктом дослідження є програмування в шкільному курсі інформатики 8 класу.

Предметом дослідження є методика навчання програмуванню учнів 8 класу.

Відповідно до мети були поставлені такі **задачі**:

1. Вивчити наукову, методичну літературу з програмування та методики його навчання в загальноосвітній школі:
   - Провести аналіз документів та підручників з інформатики на тему: «Алгоритмізація та програмування»
   - Провести аналіз існуючих авторських підходів до навчання програмуванню учнів середніх класів.
   - Провести порівняльний аналіз мов програмування Pascal та Python, також визначити переваги мови програмування Python при її використанні в шкільному навчальному процесі.
2. Розглянути методичні особливості навчання мові програмування Python.
3. Створити власні розробки уроків з даної теми.
4. Провести аналіз отриманих даних та зробити висновки.

1 РОЗДІЛ ПРОГРАМУВАННЯ В ШКІЛЬНОМУ КУРСІ ІНФОРМАТИКИ

1.1 Аналіз програм, підручників в шкільному курсі інформатики

Проаналізувавши навчальну програму було виявлено, що в саме процесі навчання інформатики учень має опанувати такі вміння:

- визначати послідовність дій, які необхідно виконати для розв’язування певних задач, тобто розробляти *алгоритми;*
- подавати алгоритми в певному формальному вигляді та виконувати їх;
- застосовувати алгоритми для опрацювання різнотипних повідомлень;
- добирати якомога ефективніший алгоритм розв’язування задачі (на зазначених уміннях базується *алгоритмічне мислення*);
- визначати параметри об'єктів та їх можливі значення;
- класифікувати явища та об'єкти;

Час, що необхідний для досягнення цих результатів, визначається вчителем залежно від рівня попередньої підготовки учнів, обраної методики навчання, наявного обладнання тощо. Однак на опанування тем змістової лінії «Моделювання, алгоритмізація та програмування» має приділятися не менше 40% навчального часу в 5–8 класах і не менше 30% у 9 класі. За необхідності вчитель може змінювати порядок вивчення тем, не порушуючи змістових зв’язків між ними.

Програмою не обмежується використання вчителем різних видів апаратного та програмного забезпечення за умови відповідності його вимогам даної програми.

Очікувані результати навчання та зміст навчального матеріалу

Учень/учениця ***Знаннєва складова*** *Розуміє* призначення мови програмування та основних її елементів. Наводить приклади сучасних мов програмування. *Знає* відмінність між змінними та константами. *Порівнює* особливості різних середовищ програмування. *Розуміє* поняття об'єкта в мові програмування, його властивостей і методів. *Пояснює* структуру програми. *Пояснює* функції елементів графічного інтерфейсу та користується ними. *Розрізняє* властивості і методи елементів управління ***Діяльнісна складова*** *Планує* процес розв'язування задачі з використанням програмування. *Створює і налагоджує* програми, зокрема подійно- й об'єктно-орієнтовані. *Використовує* в програмах вирази, коректно добирає типи даних. *Розв'язує* задачі з використанням усіх базових алгоритмічних структур, змінних та констант.	Сучасні мови програмування. Поняття об'єкта в мові програмування, його властивостей і методів. Графічний інтерфейс, основні компоненти програми з графічним інтерфейсом. Поняття елемента керування. Обробники подій, пов'язаних з елементами керування. Властивості та методи елементів керування. Типи даних у програмуванні. Структура програми. Введення й виведення даних. Вирази. Логічні вирази та змінні й операції над ними. Умовні оператори (коротка та повна форма). Складені умови. Оператори циклу. Вкладені цикли. Пошук найбільшого та найменшого серед кількох значень

Обґрунтовує вибір типів даних для розв’язування задачі ***Ціннісна складова*** *Оцінює* відповідність результатів виконання програми поставленій задачі. *Розпізнає* задачі, для розв’язання яких доцільно використовувати засоби програмування	

1.2 Платформи для вивчення алгоритмізації

Сьогодні в освіті простежується тенденція підвищення значення навчання тем курсу інформатики, пов'язаних з алгоритмізацією і програмуванням, що відбувається через затребуваність в сучасному інформаційному суспільстві фахівців в IT-індустрії і наукомістких підприємствах [3]. Розділ «Алгоритмізація та програмування» грає найважливішу роль в процесі розвитку інтелекту та інших загальних компетенцій у школярів. Проте варто зауважити:

1) кількість годин на вивчення недостатньо

2) не можна не відзначити труднощі сприйняття учнями складних алгоритмічних конструкцій і основних понять всього розділу. Незважаючи на важливість курсу інформатики і знань, пов'язаних із зазначеними темами, на сьогоднішній день

3) простежується зниження інтересу у школярів до вивчення програмування.

При навчанні алгоритмізації і програмування в 5-7 класах перед вчителями досі існує проблема навчально-методичної підтримки навчання за даними темами. Це зазвичай вирішують в двох аспектах: продовжити більш поглиблене вивчення програмних продуктів, що розглядаються в початковій школі (наприклад, «Сходинки») або почати вивчати мови програмування,

призначені для старших класів (наприклад, "Python", "Pascal", "Delphi" і ін.) [4, 5].

Багато дослідників відзначають, що альтернативою є візуальні середовища програмування, які не вимагають високого професіоналізму в області програмування, значно скорочують час на розробку програми, дозволяють враховувати вікові особливості учнів [1, 3, 6, 7].

Під візуальними середовищами розробки комп'ютерних ігор розуміють ігрові рушії, які є центральними програмними компонентами ігор та забезпечують основні технології для запуску ігор, взаємодії з операційною системою тощо, а також конструктори ігор, з вбудованими ігровими рушіями і візуальним програмним інтерфейсом [7]. У даних середовищах, є можливість, наприклад, за допомогою візуального редактора створити гру без написання тексту програми на мові програмування. Текст програми створює конструктор автоматично в візуальному редакторі, його можна при необхідності дивитися і змінювати, що дозволяє вбудувати такі візуальні середовища програмування в навчальний процес.

Розробляючи самостійно комп'ютерні ігри у візуальних середовищах, учні набувають і вдосконалюють уміння і навички програмування в процесі гри, що значно підвищує мотивацію, коли вони наділяють створених персонажів гри певними властивостями, програмують їх дії, при цьому використовуючи алгоритмічні конструкції, які при традиційному навчанні сприймались би дуже складно.

В якості найбільш відомих і популярних середовищ можна привести: Alice (www.okindiansbc.org); Kodu (https://www.kodugamelab.com), Scratch (http://scratch.mit.edu), LightBot (www.gameroo.nl/games/light-bot); StencylWorks (www.stencyl.com); Game Editor 1.40 (game-editor.com); Squeak (www. squeakland.org) та інші. [8, 3, 4].

Потрібно відзначити, що набирає все більшу популярність в освітньому процесі об'єктно орієнтоване середовище програмування "Scratch", в якій

програмні блоки збираються з цеглинок-команд, розділених в групи за кольором і своєму призначенню відповідно.

Середовище "Scratch" влаштоване таким чином: в ній можна управляти різними об'єктами, легко їх видозмінювати, переміщувати по екрану, задавати форму взаємодії між ними та ін. Головний принцип складання програм - збирання її з графічних блоків, що знаходяться в спеціальних розділах по їх загальним призначенням [2, 4]. Необхідно відзначити, що середовище має всі необхідні технічні можливості: можливість об'єктно мови програмування високого рівня, наявність графічного редактора, вбудованих інструментів для створення музики, системи допомоги користувачам, готових проектів інших користувачів та інше.

Розглянемо основні можливості "Scratch" і рекомендації вчителю щодо створення завдань учням. В даний момент середовище не підтримує процедури, функції і рекурсію, тому навчання програмування в старшій школі було б досить обмеженим. Однак технічних характеристик середовища досить для демонстрації теоретичного матеріалу, практичних робіт та інших організаційних форм уроку з інформатики для учнів 5-7 класів.

Навчання програмуванню в школі важливо проводити, використовуючи приклади типових завдань, в яких поступово ускладнюється структура алгоритмів. Це допоможе систематизувати знання учнів за класифікацією алгоритмів, необхідних в подальшому для програмування (лінійні алгоритми, розгалужені алгоритми, циклічні алгоритми). Середовище сприяє зв'язку всіх теоретичних знань алгоритмів і програмування однією практичною задачею - створенням комп'ютерної гри. Звичайно ж, в процесі створення гри учнями легко освоюються складні для сприйняття принципи об'єктно орієнтованого програмування [4].

Необхідно забезпечити заздалегідь ознайомлення з метою навчальної діяльності, створення необхідної пізнавальної мотивації в учнів. При вирішенні проблемних завдань учнями застосовується комп'ютерний експеримент, внаслідок якого формуються такі регулятивні універсальні навчальні дії, як

постановка цілі, планування, реалізація плану, прогнозування діяльності, контроль, корекція [8, 9].

Середовище дозволяє обмінятися власними розробками з учасниками спільноти авторів Scratch-проектів у всьому світі (scratch.mit.edu). Спільнота є відкритою, тобто в ньому будь-який користувач може не тільки переглядати всі матеріали, але і викладати на сервер співтовариства власний проект, зареєструвавшись в ньому.

Після закінчення вивчення розділу «Алгоритмізація» і виконання запропонованих учителем завдань, учні дізнаються, що таке алгоритм і яка його роль в сучасних системах управління; познайомляться з основними властивостями алгоритмів, способами запису алгоритмів (блок-схемами, навчальною алгоритмічною мовою), основними алгоритмічними конструкціями (слідування, розгалуження, вибір (множинний), цикл, структури алгоритмів) [1]. Розроблені завдання можна використовувати для формування описаних вище результатів і при підготовці учнів до програмування в більш складних і менш дружніх середовищах програмування в 8-9 класах.

Відзначимо, що при складанні практичних робіт на "Scratch" вчителю рекомендується дати волю творчому потенціалу учня, давати учневі певну свободу в створенні скрипта або сцени, щоб забезпечити розвиток його здібностей в плані пізнання і творчості. Підведення учителем до проблеми - дуже важливий момент при вивченні програмування і може простежуватися в логіці лабораторних робіт на "Scratch". Наприклад, що містять послідовні повторювані дії припускають підведення учнів до використання циклів. Необхідність досліджувати явища, об'єкти, які описуються за допомогою таких процесів в проблемних завданнях, вимагає освоєння інструментарію подійно-орієнтованого середовища.

Підводячи підсумки, відзначимо, що "Scratch" є актуальним і популярним в освіті середовищем візуального програмування. Її візуальні можливості можна найкращим чином використовувати при навчанні школярів темам «Алгоритмізація та програмування». В даний час по всьому світу

спостерігається активне впровадження в освітні програми загальноосвітніх шкіл навчальних предметів наукового, технічного, інженерного і математичного напрямку.

1.3 Порівняльний аналіз мов програмування

Перед тим як зрівнювати мови програмування, розглянемо, що з себе представляє мова програмування Python.

Python – високорівнева мова програмування загального значення з динамічною строгою типізацією і автоматичним керуванням пам’яттю, орієнтована на підвищення продуктивності розробника, читабельності коду та його якості [10]. Створена Гвідо ван Россумом і вперше випущена в світ мова в 1991 році, Python підтримує структурне, об’єктно-орієнтоване, функціональне, імперативне та аспектноорієнтоване програмування [11].

Python поширюється безкоштовно та є кросплатформенним. В більшості дистрибутивів операційної системи Linux, міститься інтерпретатор Python.

Головні відмінні особливості мови програмування Python та Паскалю.

1. **Простий синтаксис**. Python замість розділових знаків або ключових слів (в Паскалі такими словами є «begin» и «end»), використовує відступи для позначення виконання блока. Програми, написані в один рядок або з іншими помилками в структурі, не можуть бути виконаними в Python. Така особливість дозволить зменшити розмір коду та збільшити читанність коду програми. Синтаксис Python привчить учнів писати «красивий код», що покращить написання та розуміння коду. Так, наприклад, відрізняється запис циклу на Паскалі та Python.

Порівняння синтаксису циклу з передумовою в Паскаль і Python

Паскаль:

```
While s + n < 150 do
begin
s:=s+15;
n:=n-5
```

```
end;
writeln(n)
```

Python:

```
While s + n < 150:
 s=s+15
 n=n-5
 print(n)
```

2. **Динамічна типізація**. Python володіє динамічною типізацією. Це означає, що змінна зв’язується з типом даних під час присвоювання значення, а значить немає необхідності заздалегідь оголошувати змінну. За рахунок цього можна спростити усвідомлення типів даних. Також це дає змогу запобігти плутанині при розгляді різних довжин символьних, рядкових і цілочисельних типах. Також позитивним моментом в цьому контексті є оптимізація коду.

Порівняння синтаксису оголошення змінних в Паскаль і Python

Паскаль:

```
var s, n: integer;
begin
  s:=0;
  n:=75;
  end.
```

Python:

```
s=0
  n=75
```

3. **Компактний код**. Одне з очевидних переваг мови. Python – компактність програмного коду. Приклад, розв’язання задачі – поміняти місцями значення двох змінних – на мові Паскаль вирішується в три оператори [12], в Python в один рядок.

Порівняння синтаксису переприсвоювання змінних в Паскаль і Python

Паскаль:

```
c:=a;
```

```
a:=b;
b:=c;
```

Python:

```
a, b = b, a
```

4. **Високорівневі типи даних**. Python, будучи мовою дуже високого рівня, має вбудовані типи даних високого рівня, такі як динамічні масиви (списки) і словники [13]. В мові Python немає масивів у звичному розумінні цього терміну, але зате є списки, котрі можна зчитувати розширенням поняття “динамічний масив”. Ми можемо окремо працювати з кожним елементом списку, а можемо виконувати операції з всим списком, приклад, додавати та видаляти елементи, копіювати частини списку, сортувати [12]. Розглянемо приклад на заповнювання масиву однаковими значеннями. Python справляється з цією задачею в один рядок, продублювавши масив, що складається з одного нуля.

Порівняння синтаксису заповнювання масиву в Паскаль і Python

Паскаль:

```
const n = 100;
var a:array[0..n - 1] of integer;
for i:=0 to n – 1 do
    a[i]:=0;
```

Python:

```
n=100
    a=[0]*n
```

5. **Інтерактивний режим в Python**. Python може значно зекономити час при розробці програми, оскільки не потребує компіляції [13]. Для трансляції програми в машинний код використовується інтерпретатор. Інтерпретатор може використовуватись в інтерактивному режимі, що дозволяє легко експериментувати з функціями мови, писати одноразові програми або тестувати функції під час розробки програм [13]. Інтерактивний режим підійде

для самих перших уроків навчання Python, так як забезпечує наочність процесу програмування.

Інтерактивний режим в Python

Python:

```
>>> 50 – 5 * 6
20
>>> (50 – 5 * 6) / 4
    5.0
```

На основі даних відмінностей можна зробити висновок, що синтаксис і структура основних алгоритмічних конструкцій в Python в багато чому схожа з Паскалем. Але вони приведені в досконалий вигляд, код став чистішим та коротшим, крім того, Python підтримує сучасні типи даних і необхідні функції для роботи з ними. Це робить мову зручною для першого знайомства з програмуванням, особливо для школярів, та легким для вивчення самим учителем.

2 РОЗДІЛ МЕТОДИКА ВИКЛАДАННЯ

2.1 Методичні особливості вивчення мови програмування Python

Вивчення інформатики у 8 класі загальноосвітніх навчальних закладах здійснюється за навчальними програмами для учнів 5-9 класів, які розміщено на офіційному веб-сайті Міністерства освіти і науки України за посиланням: http://mon.gov.ua/. Вчитель має право на власний розсуд розподілити кількість навчальних годин за темами, але за будь-якої траєкторії навчання на опанування теми «Алгоритми та програми» має приділятися не менше 40% навчального часу у 5–8 класах. Тому з 70 годин виділених на весь навчальний рік потрібно, відвести не менше ніж 28 годин на вивчення теми «Алгоритми та програми»[2].

Календарно-тематичне планування з тематичної лінії «Алгоритми та програми» в 8 класі

Алгоритми та програми
Інструктаж з БЖД. Знайомство з мовою програмування Python. Введення. Виведення. Оператор присвоювання. Математичні операції.
Інструктаж з БЖД. Умовний оператор if.
Інструктаж з БЖД. Урок-практикум. Прості лінійні програми.
Інструктаж з БЖД. ***Практична робота №9. Прості лінійні програми.***
Інструктаж з БЖД. Робота з цілими та дійсними числами в Python. Дії з текстом.
Інструктаж з БЖД. Підключення додаткових модулів. Модуль math.
Інструктаж з БЖД. Логічні типи даних. Дії з даними логічного типу.
Інструктаж з БЖД. ***Практична робота №10. Складання та виконання лінійних алгоритмів для опрацювання величин.***
Інструктаж з БЖД. Логічні оператори and, or, not. Розгалуження в Python.
Інструктаж з БЖД. Урок-практикум. Розгалуження. Розв’язання задач.

Інструктаж з БЖД. ***Практична робота №11. Складання та виконання алгоритмів з розгалуженням.***
Інструктаж з БЖД. Цикли мовою Python. Цикл while.
Інструктаж з БЖД. Урок-практикум. Цикл while.
Інструктаж з БЖД. Цикли мовою Python. Цикл for.
Інструктаж з БЖД. Урок-практикум. Цикл for.
Інструктаж з БЖД. ***Практична робота №12. Складання та виконання алгоритмів з розгалуженням та повторенням для опрацювання величин.***
Інструктаж з БЖД. **Інтерфейс користувача на мові** Python. Створення вікон та налаштування їх властивостей.
Інструктаж з БЖД. **Події та обробники подій. Вікно повідомлення.**
Інструктаж з БЖД. ***Практична робота №13. Створення об'єктно-орієнтованої програми, що відображає вікно повідомлення.***
Інструктаж з БЖД. **Створення кнопок та налаштування їх властивостей.**
Інструктаж з БЖД. **Написи та їх властивості.**
Інструктаж з БЖД. ***Практична робота №14. Створення програм з кнопками та надписами.***
Інструктаж з БЖД. **Текстове поле, його функції та властивості.**
Інструктаж з БЖД. **Перемикачі та прапорці.**
Інструктаж з БЖД. ***Практична робота №15. Створення програм з текстовими полями, надписати, кнопками, перемикачами та прапорцями.***
Інструктаж з БЖД. Виконання індивідуальних і групових навчальних проектів.

Python - мова з динамічною типізацією, в ній немає необхідності заздалегідь оголошувати функції і змінні для подальшого використання, що істотно скорочує програму і робить її більш наочною. Багато вчителів негативно ставляться до такого старту в вивченні основ програмування. Однак

відсутність суворої типізації дозволяє приділити більше уваги логіці програмування і керуючим структурам. Як відомо, одним з дидактичних принципів в методиці навчання є принцип наочності. за кожним досліджуваним поняттям у свідомості учня повинен закріпитися якийсь візуальний образ, тому вже на перших заняттях учні приступають до виконання практичних робіт (використання Python як калькулятора, робота в інтерактивному режимі). При знайомстві з синтаксисом мови особлива увага приділяється «правилу відступів», тому учні відразу звикнуть до хорошого стилю програмування.

Для цього вивчення алгоритмізації та програмування поділяється на два етапи, вивчення алгоритмів, а потім вже програмування. Як правило, вчителі зупиняються тільки на вивченні алгоритмізації, оскільки вона невелика кількість викладачів інформатики має відповідний рівень підготовки викладання програмування.

Вивчення цієї теми допоможе виробити алгоритмічне мислення учнів, яке саме по собі є основою для вивчення програмування. Тому вивчення алгоритмізації є важливою частиною курсу. Вивчення алгоритмізації починається із введення поняття про сам алгоритм.

В підручнику Морзе Н.В., Барна О.В., Вембер В.П. Алгоритм визначається як послідовність команд, що керують роботою об’єкта, далі дається більш строге визначення - чітка та точна інструкція для виконавця від початкових даних до кінцевого результату[14].

У підручнику Гуржій А.М., Карташова Л.А., Лапінський В.В., Руденко В.Д. визначається як сукупність інструкцій (вказівок), виконання яких у певному порядку приводить до правильного розв’язання заданої задачі із класу однотипних задач[15].

Казанцева О.П., Стеценко І.В. у своєму підручнику алгоритм представили як чіткий опис послідовність дій[16].

У повсякденному житті діти не стикаються з цими поняттями дослівно, але алгоритми прослідковуються в різних видах діяльності

людини, про що важливо повідомити дітей по перших етапах вивчення та підтвердити це прикладами. Вводячи поняття алгоритму вчитель повинен зосередити увагу учнів на тому, що алгоритм завжди складається з орієнтацією на виконавця алгоритму.

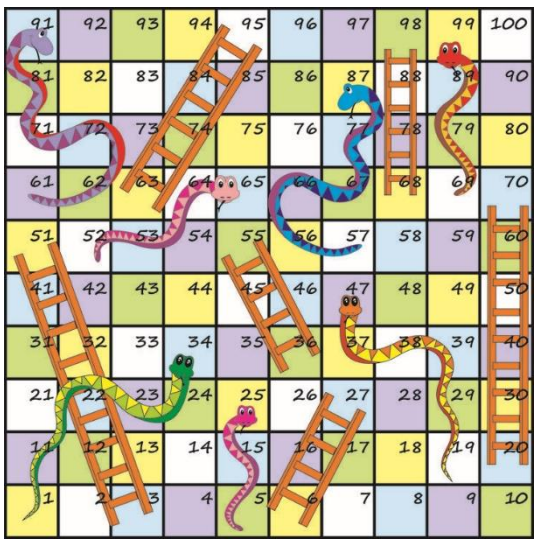
Однією з особливостей курсу є "Алгоритмізація та програмування" є практична спрямованість, тому необхідно вводити поняття алгоритму на основі практичних прикладів з повсякденного життя учнів. Школярі самі повинні виступати як виконавці простих алгоритмів, наприклад малювання кола циркулем.

Залежно від того, який клас вивчає певну тему задачі, для виконавців можуть бути складніші: знайти корінь квадратного рівняння, побудувати вписане в трикутник коло тощо.

Важливим є розвиток вміння учнями використовувати допоміжні алгоритми якомога раніше, вже на прикладах лінійних алгоритмів. Потім слід вивчати цикли. Для складання циклічних алгоритмів, спочатку необхідно теоретично підготувати учнів. Потрібно досконало розібрати циклічні алгоритми за допомогою блок схем і алгоритмічної мови. Лише тоді переходити до практики, інакше діти можуть не засвоїти цикли належним чином та діяти по прикладах, не замислюючись над змістом завдання.

Нарешті, вивчення основних алгоритмічних структур завершується розгалуження. Успішність учнів у засвоєнні цієї теми в більшості залежить від набутих раніше загальноосвітні навичок. Без сумніву, навички, які складають основу алгоритмічного мислення, слід розробляти з початкових класів.

Алгоритми можуть бути простими, складними, проте у всіх них є спільні



риси. Ось за цими характеристиками і прийнято виділяти три типи алгоритмів, з якими слід познайомити учнів. В алгоритмах команди записуються один за одним в певному порядку. Виконуються вони не обов'язково в записаній послідовності. Можуть існувати внутрішні посилання до різних команд. Взагалі, виконання команд за алгоритмом чимось нагадує настільні ігри, в яких учасники по черзі кидають кубики і ходять по полях. Причому на полях можуть бути коментарі в стилі: «Поверніться на 2 клітинки назад» або «Пройдіть на 5 клітинок вперед».

Обов’язково учнів потрібно знайомити з типами алгоритмів починаючи з лінійних алгоритмів, вони є простішими в порівнянні з іншими. Так як всі дії виконуються однократно в заданому порядку і строго послідовно. Отже, лінійний алгоритм – це алгоритм, який послідовно описує, дії які будуть виконуватися.

При вивченні лінійних алгоритмів можна запропонувати ще таке завдання. Складіть лінійний алгоритм дзвінка по мобільному телефону.

Відповідь:

1. Відкрити список контактів.
2. Обрати необхідний номер телефону.
3. Натиснути кнопку виклику.
4. Дочекатися з’єднання.

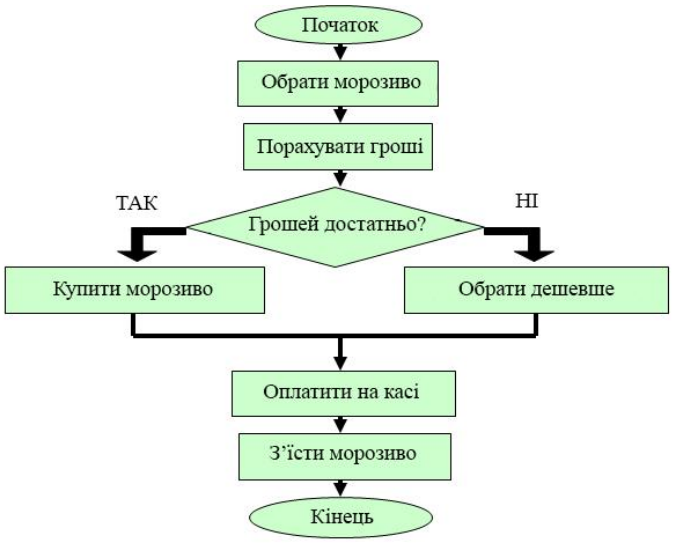
Також розглянемо простий приклад лінійного алгоритму. Алгоритм «Відчини двері».

Рідко в нашому житті зустрічаються ситуації, коли відома точна послідовність дій. Часто ми стоїмо перед вибором і приймаємо рішення в залежності від ситуації. Якщо на вулиці світить сонце, то парасольку і дощовик залишимо вдома, інакше все це візьмемо з собою. Але вибір не завжди буває таким простим.

Алгоритм з розгалуженням – це алгоритм, в якому в залежності від деякої умови виконується одна або друга дія.

Логіку прийняття рішення можна описати так: ЯКЩО <умова> ТО <дія 1> ІНАКШЕ <дія 2>.

Розглянемо приклад алгоритму «Купи морозиво».

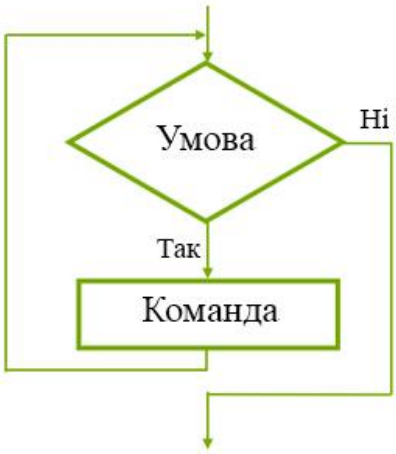


Часто в задачах одна або декілька дій необхідно повторювати декілька разів, поки не виконається наперед задана умова.

Алгоритм з повторенням або цикл – це алгоритм, в якому одна або декілька дій повторюються до тих пір, поки не буде виконана умова.

Зациклювання – це безкінечне повторювання однієї й тієї ж дії, тобто виконання циклу ніколи не закінчується.

Графічно алгоритм з повторенням можна зобразити таким чином:



Якщо умова виконується, то ми рухаємося за стрілкою «Так» і виконуємо ряд команд, після цього умова перевіряється до тих пір, поки вона не перестане виконуватися, тоді за стрілкою «Ні» можна буде переходити до наступних дій.

Учням можна запропонувати таке завдання з циклом. Складіть блок-схему в якій потрібно заповнити корзину яблуками.

Поточний контроль успішності школярів може здійснюватися у формі гри з захистом та обговоренням виконаних завдань з їх оцінкою - аналізом. Обговорення відбувається на практичному занятті. Дуже часто в подальші роботі, проаналізовані помилки дають ефективні результати правильного розуміння матеріалу і в учнів формуються правила виконання подібних завдань.

У Python є вбудовані типи: булеві, рядки, Unicode-рядки, цілі числа довільної точності, числа з плаваючою комою, комплексні числа і деякі інші. З колекцій Python підтримує кортежі (tuples), списки, словники (асоціативні масиви) і, починаючи з версії 2.4, множини. Всі значення в Python є об'єктами, в тому числі функції, методи, модулі, класи. В темі «Типи даних в програмуванні» учні знайомляться з трьома типами даних: цілі числа (integer), числа з плаваючою точкою (float), рядки (string). Оператори while, if, for містять оператори переміщення. В операторі for відбувається порівняння змінної і списку. Щоб отримати список цифр до числа <number> - використовуйте функцію range (<number>). Ось приклад використання операторів:

```
N= input ( ‘ Кількість □ елементів : □ ’ )
S=0
for i in range (N-1) :
   a=input ( ‘ Введіть □ число : □ ’ )
   S=S+a
C=S/N
print ‘Результат :’, C
```

Після розгляду операторів управління і циклу учні знайомляться зі структурами даних: рядки, списки, словники. Списки - схожі на одномірні масиви (але можна використовувати Список, який включає списки - багатовимірний масив), кортежі - незмінні списки, словники - теж списки, але індекси можуть бути будь-якого типу, а не тільки числовими. "Масиви" в Python можуть містити дані будь-якого типу, тобто в одному масиві можуть перебувати числові, строкові та інші типи даних. Масиви починаються з індексу 0, а останній елемент можна отримати за індексом -1. Для оголошення функції в Python служить ключове слово «def». Аргументи функції задаються в квадратних дужках після назви функції. Можна задавати необов'язкові аргументи, привласнюючи їм значення за замовчуванням

Програми на Python істотно лаконічніше Pascal, що істотно полегшує завдання знайомства з мовою початківцям програмістам, так як пошук помилок

та налагодження вимагає істотно менших витрат часу. Порівняємо, наприклад, два елементи коду програми, написаних на мовах програмування Pascal і Python.

Python:

a = [1] * 1000

Паскаль:

var a: array [1..1000] jf integer;

...

for i: = 1 to 1000 do

a [i]: = 1;

Виходячи з наведеного коду можна побачити, що на двох мовах програмування записані рівнозначні операції, в результаті виконання яких ми отримаємо масив з 1000 елементів, заповнених одиницями. Однак на Python цей код займає 1 рядок, тоді як на Pascal 3.

Можливо, з методичної точки зору, при вивченні даного розділу і вирішенні наведеної вище завдання, школяреві необхідно пояснити, що масив є безперервним фрагментом виділеної пам'яті, і при його створенні ми повинні зарезервувати під нього в пам'яті місце, оголосивши його, а потім ініціювати. Однак рядок a = [1] * 1000, на наш погляд, відображає зміст виконуваного школярем дії (потрібен масив з числа 1 повтореного 1000 разів) повніше і, в підсумку, простіше в написанні.

Розглядаючи питання навчання інформатиці в 7-9 класах можемо відзначити що ось ці початкові уявлення про програмування, можливо і є поріг на якому дитина зупиниться, отримавши загальне уявлення про масивах, їх оголошенні і обробці.

Наступним ступенем навчання, в профільних класах, вчитель і учень отримають в своє розпорядження досить універсальну, сучасну мову програмування, яка реально застосовується для розробки програмного забезпечення.

Безумовно, велика кількість високорівневих підпрограм вбудованих в мову Python, широкий функціонал мови, буде приводити до того, що перед школярем виникне спокуса використання цих можливостей, замість реального вивчення алгоритмів і принципів роботи цих функціональних елементів. Однак тут необхідно розглянути методичну особливість вивчення такої мови програмування, пов'язану з тим, що школярі повинні спочатку вивчити принципи і алгоритми роботи окремого функціонального елементу, і тільки потім перейти до його використання при вирішенні задач. Наприклад, вирішення наступного завдання: записати значення змінної a в змінну b. завдання повинна спочатку бути вирішена шляхом використання додаткової змінної c, в яку ми запишемо значення перезаписуваної змінної a:

c = a

a = b

b = c

і тільки після цього можливо використовувати кортежі мови Python:

(A, b) = (b, a).

Таким чином викладач може підійти до вирішення більшості завдань, пов'язаних з сортуванням масивів, пошуком елементів, що в кінцевому підсумку на профільному етапі навчання дозволить школяреві вирішувати більшу кількість різноманітних завдань в стислі терміни.

В результаті можна відзначити, що не тільки немає необхідності відмовлятися від ідеї вивчення високорівневих мов програмування в школі, а навпаки, вивчення Python, при правильному підході і обліку методичних особливостей, відкриє перед учнем нові горизонти і можливості, так як сучасні мови програмування, вдосконалюючись, стають все більш універсальними, гнучкими і простими, зручними для сприйняття і налагодження. Такий підхід до вивчення високорівневих мов дозволить готувати вже на шкільній ступені початківців програмістів, що мають різносторонній досвід в написанні програм.

2.2 Розробки уроків

Урок №1

Клас: 8

Тема уроку: Знайомство з мовою програмування Python. Введення. Виведення. Оператор присвоювання. Математичні операції.

Мета уроку:

1) *Навчальна:* ознайомити з мовою програмування Python, навчити користуватися функціями введення та виведення, ввести поняття змінної і оператора присвоювання, ознайомити з математичними операціями.

2) *Розвиваюча:* розвиток мислення, мови, алгоритмічного стилю мислення.

3) *Виховна:* виховання емоційно-позитивної направленості на практичну діяльність, інтересу до інформатики, особистої відповідальності за результати своєї роботи.

Тип уроку: комбінований.

Обладнання: комп’ютери, проектор, середовище Python IDLE, презентація.

Хід уроку

1. **Організаційний етап** (2 хвилини).

- привітання
- облік відсутніх

2. **Вступна частина** (5 хвилин)

Сьогодні ми почнем велику нову тему. Програмування на мові Python. Дайте відповідь на питання, що таке програмування?

Програмування – це створення комп’ютерних програм. Всі програми: ігри, антивіруси, текстові редактори на комп’ютері були написані програмістами. Ми з вами, звісно, не зможемо створити таку велику програму як антивірус або редактор Microsoft Office Word, але будемо намагатися створити маленькі ігри.

Комп'ютерні програми пишуть на спеціальних мовах програмування. *Мова програмування* – це мова, зрозуміла комп'ютеру. В даний час мов програмування дуже багато. Хто може назвати які-небудь мови програмування?

Найпоширенішими мовами програмування зараз є Python, Java, JavaScript, C#, C, C++, PHP, SQL, Ruby.

Ми будемо вивчати програмування на мові Python. Це сучасна мова, вона постійно розвивається, допрацьовується. Ця мова використовується в таких проектах, як Google, YouTube, Instagram, Facebook та інших. Він легкий в освоєнні та простий у використанні.

3. **Практична робота за комп'ютерами** (20 хвилин)

Програми пишуться в спеціальних середовищах програмування. Відкриємо середовище програмування Python:

Пуск –> Python 3.9 –> IDLE (Python GUI) –> File –> New File

Отже, давайте напишемо першу програму, яка виведе повідомлення «Hello, world!»

Для цього достатньо набрати наступний код:

```
print ("Hello, world!")
```

print – функція (команда) виводу.

Запис в зошит:

Функція виводу:

```
print ("текст")
```

Друге, що ми вивчимо – це змінну та оператор присвоювання. (Пишемо нову програму).

```
message = "Hello, world!"
print (message)
```

Змінна – це величина, яка має ім'я, тип та значення. Значення змінної можна змінювати під час роботи програми. В програмі ми створили змінну з ім'ям message, присвоїли їй значення-рядок "Hello, world!" і відповідно ця змінна приняла рядковий тип.

Знак «=» - це оператор присвоювання.

Імена змінних можуть складатися з:

- Латинських букв (маленькі та великі букви різняться!)
- Українські, російські (не рекомендуються)
- Цифри (ім’я не може починатися з цифри і складатися тільки з цифр)
- Знак підкреслення _

Не можна використовувати в іменах змінних:

- Пробіли
- Знаки +, -, >, <, =, (,), ! та інших
- Ключові слова мови Python

Не можна використовувати як назви змінних ключові слова мови Python:

- False
- None
- True
- and
- as
- asset
- break
- class
- continue
- def
- del
- elif
- else
- except
- finally
- for
- from
- global
- if
- import
- in
- is
- lambda
- nonlocal
- not
- or
- pass
- raise
- return
- try
- while
- with
- yield
- print

Ключові слова - це слова мови програмування, які мають спеціальне, раз і назавжди закріплене за ними значення. До них відносяться імена функцій, операторів та інших. Наприклад, функція «print» - ключове слово, яке не можна використовувати в якості імені змінної. Пізніше ми вивчимо і інші функції.

Перейдемо до знайомства з математичними операціями. (Створюємо новий файл).

Створимо дві цілочисельні змінні й дамо команду комп’ютеру додати їх.

```
a = 78001457
b = 2546880
c = a + b
print (c)
```

Змінною с можна присвоїти цілий математичний вираз:

```
c = (a-b) * (a+b) / 27
```

Інші математичні операції:

x + y	Додавання
x - y	Віднімання
x * y	Множення
x / y	Ділення
x // y	Ціла частина від ділення
x % y	Остача від ділення
-x	Зміна знаку числа
abs(x)	Модуль числа
divmod(x, y)	Пара (**x // y, x % y**)
x ** y	Піднесення до степеня

Функції введення.

Для того щоб присвоїти змінній значення, введене з клавіатури, виконується функція input (). Напишем і запустим наступну програму:

```
name = input(“Введіть своє ім’я: “)
print(“Привіт, “, name)
```

Змініть програму так, щоб вона виводила в кінці знак оклику.

Запис в зошит:

Введення рядка:

```
s = input (“Введіть рядок: ”)
```

“Введіть рядок: ” – звернення до користувача (не обов’язково, але бажано)

Стандартно всі введені дані інтерпретатор Python розуміє, як рядки, тому, якщо ми хочемо отримати число, то рядок прийдеться перетворити в число.

Перетворення до цілочисельного типу та введення цілого числа:

Запис в зошит:

Введення цілого числа:

```
n = int ( input (“Введіть число: ”) )
```

Тобто на функцію введення ми навішуємо ще одну функцію перетворення в ціле число.

Запис в зошит:

Функція перетворення до цілочисельного типу:

```
n = int ( s )
```

Функція перетворення до рядкового типу:

```
s = str ( n )
```

Завдання. Напишіть програму, яка буде отримувати на введення 2 числа та виведе їх суму.

```
a = input("Введіть число a: ")
b = input("Введіть число b: ")
sum = a+b
print(“a+b= “, sum)
```

Чому програма працює не правильно? (Тому, що всі введені дані комп’ютером сприймаються як рядок (набір символів, не цифр)) Що виправити в програмі, щоб вона працювала правильно?

Правильний варіант:

```
a = int(input("Введіть число a: "))
b = int(input("Введіть число b: "))
sum = a+b
print(“a+b= “, sum)
```

Задача. В кожному рядку визначити тип і значення змінної.

```
a = 5
n = input()        #користувач вводить цифру 8
c = int(n)
d = a*c
d = d–a
s = “Рамамбахарумамбуру”
d = n+a
m = n+s
```

Запис в зошит:

Коментар до програми, комп’ютер їх не читає.

Генератор випадкових чисел

Запис в зошит:

Функція генерації випадкового цілого числа з відрізка [x, y]:

```
Import random
a = random.randint (x, y)
```

4. **Самостійна робота на комп’ютерах** (13 хвилин)

Учні намагаються самостійно вирішити задачі:

1) Вивести на екран три введених з клавіатури числа в порядку, оберненому їх вводу.

2) Ввести з клавіатури два числа та вивести цілу частину від ділення першого на друге.

3) Ввести з клавіатури основу та висоту трикутника і вивести площу трикутника.

4) Ввести з клавіатури два катета трикутника і вивести гіпотенузу. (Квадратний корінь – це піднесення до степеня (1/2))

5) Згенерувати випадкове двозначне число, вивести на екран це число, також суму та добуток його цифр.

Для отримання цифр використовуйте цілочисельне ділення на 10 та візьміть остачу від ділення на 10. Приклад для числа 47.

47//10=4

47%10=7

5. Оцінювання роботи учнів на уроці.

6. Домашнє завдання

Встановити на комп'ютер середовище програмування IDLE. (Завантаження тільки з офіційного сайту https://www.python.org/)

Написати програми:

1) Ввести основу та висоту трапеції та вивести площу трапеції.

2) Отримати випадкове тризначне число, вивести це число та суму його окремих цифр.

3) Програма, яка розраховує вік людини в годинах.

Урок №2

Клас: 8

Тема уроку: Умовний оператор if, else, elif.

Мета уроку:

1) *Начальна:* ознайомити учнів з умовним оператором та навчитися застосовувати його при створенні програм на мові програмування Python.

2) *Розвивальна*: розвиток мислення, мови, алгоритмічного стиля мислення.

3) *Виховна:* виховання емоційно-позитивної направленості на практичну діяльність, інтересу до інформатики, особистої відповідальності за результати своєї роботи.

Тип уроку: комбінований.

Обладнання: комп'ютери, проектор, середовище Python IDLE, презентація.

Хід уроку

1. **Організаційний етап** (2 хвилини)

- привітання
- облік відсутніх

2. **Перевірка домашнього завдання** (10 хвилин)

Задане на минулому уроці домашнє завдання:

Встановити на комп’ютер середовище програмування IDLE.

Написати програми:

1) Ввести основу та висоту трапеції та вивести площу трапеції.

2) Отримати випадкове тризначне число, вивести це число та суму його окремих цифр.

3) Програма, яка розраховує вік людини в годинах.

3. **Теоретична частина** (10 хвилин)

Відкрийте зошити та запишіть тему уроку: «Умовний оператор if».

На минулому уроці ми навчились складати лінійні програми на мові програмування Python. Сьогодні ми вивчимо конструкцію «розгалуження» або «умовний оператор if».

Якщо перевести на українську мову, конструкція умовного оператора означає наступне:

Якщо <виконується умова> виконати: якісь дії.

Наприклад:

```
if a > b:
    print ( a )
```

«Якщо a більше b, то вивести a».

Або:

```
if x = = y :
    z = x + y
  z = z * z
```

«Якщо x рівний y, то z присвоїти значення x + y та піднести z в квадрат».

Відступи важливі! Вони – частина коду. Дії будуть виконуватись тільки тому випадку, якщо всі вони записані з відступами, і при чому з однаковою кількістю відступів. Стандартно в Python-спільноті прийнято робити 4 відступи.

Загальна форма запису неповної формули умовного оператора:

If <умова>:

    <дія 1>

    <дія 2>

    і так далі

Задача. Що буде надруковано в результаті роботи програми?

```
a=7
b=9
if a>b:
    print(a)
```

(Відповідь: нічого)

Це була неповна форма умовного оператора. Но в умовного оператора також є і повна форма. Українською мовою вона звучить так:

Якщо <виконується умова>: виконувати якісь дії. Інакше: виконувати інші дії.

Інакше означає «якщо умова не виконується».

Наприклад:

```
if a > b :
    print ( a )
else :
    print ( b )
```

«Якщо a більше b, то вивести a, інакше вивести b.

Загальна форма запису неповної форми умовного оператора:

if <умова>:

    <дія 1>

else:

<дія 2>

Задача. Що буде надруковано в результаті роботи програми?

```
a=8
b=5
 if a<b:
    print(a)
else:
    print(b)
```

Часто зустрічаються задачі з великою кількістю умов та дій, які потрібно провести при виконанні цих умов. Конструкції if-else не достатньо, тоді на допомогу приходить оператор elif. Українською мовою він пояснюється так:

Якщо <виконується умова 1>: виконувати якісь дії.

Інакше якщо <виконується умова 2>: виконувати другі дії.

Інакше якщо <виконується умова 3>: виконувати треті дії.

Інакше: виконувати щось ще.

Останнє «інакше» означає «якщо ніякі з вище перелічених дій не виконуються». Присутність «інакше» не обов’язково. Наприклад:

```
cost = 1500
if cost < 1000:
    print ( "Знижок немає" )
elif cost < 2000:
    print ( "Знижка 2%" )
elif cost < 5000:
    print ( "Знижка 5%" )
else:
    print ( "Знижка 10%" )
```

Що буде надруковано в результаті роботи програми?

(Відповідь: Знижка 2%)

Запис в зошит:

Знаки відношень:

> більше >= більше або рівне

< менше <= менше або рівне

= = дорівнює != не дорівнює

Запис в зошит:

Складні умови.

Щоб скласти складну умову виконуються оператори:

and – «і»

or – «або»

not – «не»

Приклад:

```
If a > 0 and a < 10 or a = = 100:
print ( a )
```

Чи буде надруковано а, якщо а рівне 7? А якщо а рівне 20?

Пріоритет:

1) Відношення (<, >, <=, >=, = =, !=)
2) not («НЕ»)
3) and («І»)
4) or («АБО»)

4. Робота на комп'ютерах (20 хвилин)

Учні пишуть програми на комп'ютерах під керівництвом вчителя.

Задачі:

1) Ввести ціле число. Якщо це число більше 5, то вивести повідомлення: «Це число більше 5».

2) Ввести ціле число. Якщо воно є додатнім, то додати до нього 1; в протилежному випадку відняти від нього 2. Вивести отримане число.

3) Перевірити, чи належить число, введене з клавіатури, інтервалу (-9;2).

4) Написати програму «Провидець». Програма повинна запропонувати користувачу ввести питання, на який можна відповіти однозначно, тобто «так» або «ні». Після чого користувачу випадковим чином видається відповідь,

наприклад: «Так», «Ні», «Безперечно, так!», «Ні в якому разі!», «Звісно ж ні!, досить задавати дурні запитання!» і тому подібні. Варіантів відповідей повинно бути не менше чотирьох.

5) Ввести число a. Визначити та вивести повідомлення про те, парне чи не парне воно. Для визначення парності використовуйте лишок від ділення на 2: якщо a % 2 = = 0, то a – парне.

6) Визначити, чи є трикутник зі сторонами a, b, c рівнобедреним.

7) По номеру дня тижня вивести його назву.

8) Дані цілочисельні координати точки на площині. Якщо точка співпадає з початком координат, то вивести 0. Якщо точка не співпадає з початком координат, але лежить на осі OX або OY, то вивести відповідно 1 або 2. Якщо точка не лежить на координатних осях, то вивести 3.

5. Оцінка роботи учнів на уроці.

6. Домашнє завдання (3 хвилини)

Написати програми:

1) Дано ціле число. Якщо воно є додатнім, то помножити його на 3; в протилежному випадку відняти від нього 100. Вивести отримане число.

2) Визначити, чи є число a дільником числа b.

3) Визначити можливість існування трикутника по сторонам. (Трикутник існує тоді, коли сума будь-яких 2 його сторін більше 3).

2.3 Впровадження розробок в освітній процес

Під час проходження навчальної практики в загальноосвітній школі, ці роботи були запропоновані в підгрупу на заняття з інформатики. Але мені як студенту-практиканту було відмовлено на це були такі причини:

- Закріплена підгрупа 8 класу почала вивчати середовище розробки програмного забезпечення Lazarus мові Object Pascal. І як наслідок, було вже не доцільно змінювати начальний план.

- Протягом всієї практики часто діти переходили на дистанційне навчання, при якому таку тему було б складно подати доступно учням.

ВИСНОВКИ

В цьому курсі були розглянуті методичні особливості навчання програмування в шкільному курсі інформатики 8 клас в якості інструмента використовувалась мова програмування Python. Були опрацьовані вбудовані типи даних, розгалуження та цикли. Створені розгорнуті конспекти уроків з даної теми в яких представлено різноманітні задачі. Саме на практичну складову було акцентовано увагу, адже учням цікавіше ознайомлюватись з матеріалом в процесі виконання завдань. Або ж після усної подачі матеріалу, необхідно запропонувати учням практичні завдання з теми уроку. Таким чином можна не втрачати зацікавленість учнів в програмуванні на Python.

Виявляється, що при якісній підготовці учнів до вивчення даної мови програмування, їм вдається легше засвоювати необхідний матеріал. Таку підготовку може забезпечити середовище та динамічна візуальна мова програмування як Scratch. З його допомогою можна програмувати прості інтерактивні інтерфейси, створювати мультфільми та відеоігри. Завдяки цьому в учнів виникає інтерес до програмування, що покращує успішність серед учнів

В процесі дослідження всієї теми нами була досягнута мета та виконані поставлені задачі

Python є мовою програмування високого рівня. Дана мова останнім часом стає все більш затребуваною не тільки у програмістів, але також серед студентів і школярів. У свою чергу це може стати поштовхом для вивчення цієї мови програмування в школі, не тільки в старших класах, але в середній ланці і він може повноцінно претендувати на перше місце серед тих, хто вивчає програмування в школі. Завдання вчителя інформатики загальноосвітньої школи вибрати для своєї роботи Найбільш зручні навчально-методичні комплекси і мову програмування. Це гарантує більш якісне вивчення навчального матеріалу, освоєння мов. Правильний вибір вчителя допоможе йому не тільки якісно виконувати свою роботу, а й зможе домогтися прихильності учнів.

Таким чином, Python можна і потрібно використовувати в якості базової мови початкового навчання програмуванню у 8 класі. Це дозволить підвищити інтерес до вивчення алгоритмізації та основ програмування у більш широкого кола школярів, ніж зараз, отримати більш високі освітні результати, що в кінцевому підсумку сприятиме формуванню у школярів цифрових компетенцій, необхідних для успішної професійної самореалізації в епоху цифрової економіки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Особливості вивчення програмування на Scratch О.М. Дудка, О.О. Власій Режим доступу: http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&P21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/Kitonv_2017_26_17.pdf

2. Міністерство освіти і науки України Інофрматика 5-9 класи Програма для загальноосвітніх навчальних закладів. 2017р.

3. Данильчук Е.В., Куликова Н.Ю., Гермашев И.В. Методические особенности формирования готовности будущего учителя информатики к разработке и использованию компьютерных игр в обучении алгоритмизации и программированию // Изв. Волгоград. гос. пед. ун-та. 2018. № 5(128). С. 42–49.

4. Використання програмного середевища scratch як пропедевтика до програмування Балабас, А. та Наумук, Ірина (2017) Використання програмного середевища scratch як пропедевтика до програмування. Інформаційні технології в освіті та науці: зб. наук. пр., 1 (9). с. 10-13.

5. Кирстич І.В., Василенко Я.П., Про педагогічні та дидактичні особливості Scratch як інструменту навчання основам алгоритмізації та програмування [Електронний ресурс]. – Режим доступу: http://dspace.tnpu.edu.ua/bitstream/123456789/14524/1/Kyrstych.pdf

6. Карпенко О.М. Лукьянова А.В., Абрамова А.В. [и др.] Геймификация в электронном обучении // Дистанционное и виртуальное обучение. 2015. № 4(94). С. 28–43.

7. Куликова Н.Ю. Создание и использование интерактивных компьютерных игр как средство активизации познавательной деятельности обучающихся на уроках информатики // Современные информационные технологии в образовании: сб. материалов XXVIII Междунар. конф. (Троицк-Москва, 27 июня 2017 г.). М.: Москов. издат.-полиграф. колледж им. И. Федорова. 2017. С. 27–29.

8. Быкова А.Р. Использование среды Scratch для обучения программированию учащихся основной школы // Наука молодых – будущее России: сб. науч. ст. II Междунар. науч. конф. перспектив. разработок молодых ученых: в 5 т. (г. Курск, 13–14 дек. 2017 г.). Курск: Университетская книга, 2017. С. 69–72.

9. Макарова Н.В., Нилова Ю.Н. Методика формирования навыков программирования и моделирования // Информатика и образование. 2014. № 2(251). С. 29–32.

10. Python (programming language) - Wikipedia [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Python_(programming_language)

11. Python - Вікіпедія Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Python

12. Поляков К.Ю. Язык Python глазами учителя // Информатика. 2014. № 9. С. 4-16.

13. Whetting Your Appetite – Python 3.7.3 documentation [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/tutorial/appetite.html

14. Інформатика : підруч. для 8 кл. загальноосвіт. навч. закладів / Морзе Н.В., Барна О.В., Вембер В.П. – К. : УОВЦ «Оріон», 2016. – 240с.

15. Інформатика для загальноосвітніх навчальних закладів з поглибленим вивченням інформатики : підруч. для 8кл. загальноосвіт. навч. закл. / Гуржій А.М., Карташова Л.А., Лапінський В.В., Руденко В,Д. – Львів : Світ, 2016. – 296с.

16. Інформатика : підручник для 8кл. загальноосвіт. навч. закл. / Казанцева О.П., Стеценко І.В., - Тернопіль : Навчальна книга – Богдан , 2016 – 304с.

17. Підручник мови Python / Ґвідо ван Россум, Фред Л. Дарк Молодший [Електронний ресурс]. – Режим доступу: https://docs.python.org/3/tutorial/